McMaster University
Dept. of Electrical and Computer Engineering

COE 4SL4 - Term I (Fall) 2022

Assignment 3. Binary Classification with Logistic Regression and k-Nearest Neighbours

Due Date: November 14, 11:30 pm

If you submit the assignment after the deadline, the following penalty is applied:

- 10% penalty if the submission is before November 21, 11:30 pm (if the mark before applying the penalty is 78 out of 100, after applying the penalty it is 78 - 7.8 = 70.2 out of 100);
- 50% penalty if the submission is after November 21, 11:30 pm and before Dec. 8, 11:30 pm.

Description:

In this assignment you are required to experiment with logistic regression and k-nearest neighbours for binary classification. You will use the “Wisconsin breast cancer” data set, which is available in scikit-learn. The two classes are ”malignant” (the **positive** class) and “benign” (the **negative** class) (if the classes are labeled differently, please change their labels). The number of features is 30.

After you separate the test and training data, you have to standardize the features, (i.e. center and scale them - see slide 15 in Topic 2) in the training data and then apply the transformations to the features in the test data. Then you have to do the following.

- Train a linear classifier using your implementation of logistic regression. Then compute the misclassification rate and the F1 score on the test data. Additionally, you have to plot the “precision/recall” (PR) curve (“recall” is on the horizontal axis and “precision” is on the vertical axis). For this you have to consider all possible thresholds for you classifier based on your test data and compute the precision and recall in each case (see pages 9 and 10 in Topic 6 Annotated for details).
- Do the same as above using the implementation of logistic regression provided in scikit-learn.
- Implement the k-nearest neighbour classifier for each $k = 1, 2, 3, 4, 5$, and select the best k using K-fold cross-validation (with your implementation). Then use the best classifier to compute the test error (i.e., the misclassification rate on the test data set).
- Do the same as above using the implementation provided by scikit-learn.

Whenever you use randomization in your code, use the number formed of the last 4 digits of your student ID, as seed for the pseudo number generator.

You have to write a report to present your results and their discussion. The report must include a table with the test errors for all four models. For the nearest neighbours predictors you have to also specify the cross-validation error for each k, in a table. You have to specify the value of K that you choose.
Note: In k- nearest neighbours there are situations where ties might occur.

For instance: 1) two or more training examples that are at the same distance from the current example or 2) half of the neighbours are in one class, half in the other. In such cases, you must decide how to deal with the ties and you have to document this in your report.

For the two logistic regression models you have to specify the vectors of parameters and to include a plot with the two PR curves.

How do the results obtained with your implementations compare with the results obtained using scikit-learn? Which of the four models is the best overall? To decide this, use first the misclassification rate as the metric, next use the F1 score as the metric. Is the winner the same in the two cases?

Besides the report, you have to submit your numpy code. The code has to be modular. Write a function for each of the main tasks. Also, write a function for each task that is executed multiple times (e.g, to compute the average error, to compute the K-fold cross-validation error for a model, etc). The code should include instructive comments.

You may use the following code for feature normalization:

```
import numpy as np
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Note that the normalization is performed based on the training data. Then the same transformations are applied to the test data in order to be able to use the trained predictor on the test data.

Submission Instructions:

- Submit the report in pdf format, the python file (with extension ".py") containing your code, and a short demo video. The video should be 1 min or less. In the video, you should scroll down your code, show that it runs and that it outputs the results for each part of the assignment. Submit the files in the Assignments Box on Avenue.

Instructions to obtain the precision/recall curve for a logistic regression model: The classifier that minimizes the misclassification rate is

$$\hat{C}(\mathbf{x}) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases},$$

where $z = \mathbf{w}^T\mathbf{x}$. Other classifiers can be obtained by replacing 0 by some other thresholds θ:

$$\hat{C}(\mathbf{x}) = \begin{cases} 1 & \text{if } z \geq \theta \\ 0 & \text{if } z < \theta \end{cases}.$$

To obtain all possible classifiers for the test set, it is sufficient to consider as values for θ, the values $z[0], z[1], \cdots, z[M-1]$, obtained on the test examples (where M is the size of the test set). For this, first sort the array $\mathbf{z}$. Then take in turn $\theta = z[0]$, $\theta = z[1]$, and so on, up to $\theta = z[M-1]$. For each θ compute the precision and the recall. Note that the positive class is the "malignant" class.